REMARKS

Claim 12 has been rewritten in independent form to include all of the limitations of claims 10 and 11 from which claim 12 has depended. Claim 12 is otherwise the same claim 12 as was originally filed.

The Examiner objected to the drawings.

The Examiner objected to the specification.

The Examiner objected to claims 1-9.

The Examiner rejected claims 3, 5, 14, 21 and 23 under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement.

The Examiner rejected claims 3, 5, 12, 14, 21 and 23 under 35 U.S.C. § 112, second paragraph, as being indefinite for allegedly failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

The Examiner rejected claims 1-9 and 19-27 under 35 U.S.C. § 101 because the claimed invention is allegedly directed to non-statutory subject matter.

The Examiner rejected claims 1-6, 8-15, 17-24 and 26-27 under 35 U.S.C. § 102(b) as allegedly being anticipated by <u>Hanis</u> (Hanis *et al.*, "Applying the State Pattern to WebSphere Portal Portlets", Part 1 - Overview, Part 2 - Implementation, 12/11/2002).

The Examiner rejected claims 7, 16 and 25 under 35 U.S.C. § 103(a) as allegedly being unpatentable over <u>Hanis</u> in view of <u>Hepper</u> (Hepper *et al.*, "Introducing the Portlet Specification, Part 1", 08/01/2003).

Applicants respectfully traverse the drawings objections, specification objections, claims objections, § 112 rejections , § 101 rejections , § 102 rejections, and § 103 rejections with the

10/790,913   15

following arguments.

Drawings Objections

The Examiner argues: "The drawings are objected to because the Fig.6A, 6B, the class hierarchy of class BaseController and class TemplateControllerForHtml are not consistent with Fig.4. Fig.4 shows that the class BaseController is a child of AbstractMVCPortletController and class TemplateControllerForHtml is extending from class BaseController. However, in Fig.6A and 6B, it shows "public class BaseController extends MVCPortlet" and "public class TemplateControllerForHtml extends MSCBaseController"."

In response, Applicants have submitted Replacement Sheets for Sheets 4, 7, and 8 of the drawings comprising FIGS, 4, 6A, and 6B respectively, to resolve the inconsistencies identified by the Examiner. The changes in FIGS. 4, 6A, and 6B are as follows:

in FIG 4, AbstractMVCPortletController is changed to AbstractMVCController;

in FIG. 6A, MVCPortlet is changed to AbstractMVCController; and

in FIG. 6B, MSCBaseController is changed to BaseController.

No new matter has been added.

Accordingly, Applicants respectfully request that the drawings objections be withdrawn.

Specification Objections

The Examiner argues:

"The disclosure is objected to because of the following informalities:

- Page 8, line 3, "In FIG.112" should be "In FIG.12"
- Page 13, line 19, "includes blocks 21-26", there is no block number 26 in FIG.3.
- Page 11, line 21, opened bracket."

In response, Applicants have amended the specification to clarify the invention in a manner that resolves the objections noted by the Examiner.

No new matter has been added.

Accordingly, Applicants respectfully request that the specification objections be withdrawn.

Claims Objections

The Examiner argues:

"Claims 1-9 are objected to because of the following informalities:
- Using the words "comprising: including" in claims 1-9;
- Using the words "comprises including" in claims 2-5

They appear to be misused and make the claims with grammatical and idiomatic errors."

In response, in light of the fact that amended claim 3 includes all of the limitations of claims 1-2 and that claims 1-2 have been canceled, Applicants respectfully contend that the language "comprising: including" in claim 3 should not be objected to. Applicants respectfully believe that the Examiner incorrectly considers "including" to be duplicative of "comprising" in claim 3. 7, 16 and 25 Applicants next explain that the words "including" and "comprising" have different functionalities in claim 3 and are not being used duplicatively or otherwise erroneously.

Applicants respectfully contend that "said designing the software comprising" denotes that designing the software comprises something, namely the subsequently recited steps. The first step of the subsequently recited steps is "including in the software ...", which utilizes "including" to recite an active method step beginning with "including".

In other words, the word "comprising" is coupled to "designing the software", and the word "including" is coupled to "in the software ...". Therefore, "including" and "comprising" have different functionalities in claim 3 and are not being used duplicatively or otherwise erroneously.

A similar analysis applies to "comprises including" in claims 4-5.

Accordingly, Applicants respectfully request that the objections to claims 3-9 be withdrawn.

10/790,913   19

35 U.S.C. § 112, First Paragraph

"The Examiner rejected claims 3, 5, 14, 21 and 23 under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement."

Claims 3 and 21

The Examiner argues: "The applicant claims the BasePortlet class includes the action performed method and set state method. However, the specification discloses that the BasePortlet class only includes action performed method. The set state method is defined Action class."

In response, Applicants respectfully dispute the Examiner's allegation that "the specification discloses that the BasePortlet class only includes action performed method". Applicants respectfully request that the Examiner quote the specific language in the specification that allegedly discloses that the BasePortlet class only includes action performed method.

The specification, page 15, line 20 discloses that "[the program code includes a portlet code module (BasePortlet and TemplatePortlet) ...". The specification, page 16, lines 1-2 discloses that "The portlet code module is adapted to execute the **action performed** method and the **set state method**". The specification, page 16, lines 5-6 discloses that "[the TemplatePortlet class is a child of the BasePortlet class".

Furthermore, the Action class includes both the **action performed** method (see specification, page 12, lines 9-11) and the **set state** method (see specification, page 12, lines 14-15). Thus, it is clear that the **action performed** method and the **set state** method of the Action class are accessible to the BasePortlet class comprised by the portlet code module.

Therefore, the feature of "the BasePortlet class ... includes the action performed method

and the set state method" in claims 3 and 21 is not inconsistent with the specification as alleged by the Examiner.

For further clarification, Applicants have amended the specification in the following marked up manner to specifically include the aforementioned subject matter recited in the originally submitted claims 3 and 21 (and is therefore not new matter):

> The program code includes a portlet code module (BasePortlet and TemplatePortlet) and a controller code module (BaseController and TemplateControllerForHtml) in the program code. The BasePortlet class of the portlet code module is adapted to execute the **action performed** method and the **set state method,** and the BaseController class of the controller code module is adapted to execute the **perform view** method."

Therefore, the aforementioned subject matter recited in the originally submitted claims 3 and 21 is unambiguously consistent with the specification.

Accordingly, Applicants respectfully request that the rejections of claims 3 and 21 under 35 U.S.C. § 112, first paragraph be withdrawn.

Claims 5, 14, and 23

The Examiner argues: "The applicant claims the BaseController class includes the perform view method. However, there is no description about it in the specification. The specification discloses that only the State class includes Perform view method."

In response, Applicants respectfully contends that the State class includes Perform view method, which is not inconsistent with claims 5, 14, and 23.

The specification, page 15, line 20 discloses that "[the program code includes ... a controller code module (BaseController and TemplateControllerForHtml) ...". The specification, page 16, line 2 discloses that "the controller code module is adapted to execute the **perform view** method". The specification, page 16, lines 10-11 discloses that "[the TemplateControllerForHtm is a child of the BaseController class ".

Furthermore, the State class includes the **perform view** method (specification, page 15, lines 3-4). Thus, it is clear that the **perform view** method the State class is accessible to the BaseController class comprised by the controller code module.

Therefore, the feature of "the BaseController class ... includes the perform view method " in claims 5, 14, and 23 is not inconsistent with the specification as alleged by the Examiner.

For further clarification, Applicants have amended the specification in the following marked up manner to specifically include the aforementioned subject matter recited in the originally submitted claims 5, 14, and 23 (and is therefore not new matter):

> The program code includes a portlet code module (BasePortlet and
> TemplatePortlet) and a controller code module (BaseController and
> TemplateControllerForHtml) in the program code. The <u>BasePortlet class of the</u>
> portlet code module is adapted to execute the **action performed** method and the

set state method, and the <u>BaseController class of the</u> controller code module is adapted to execute the **perform view** method."

Therefore, the aforementioned subject matter recited in the originally submitted claims 5, 14, and 23 is unambiguously consistent with the specification.

Accordingly, Applicants respectfully request that the rejections of claims 5, 14, and 23 under 35 U.S.C. § 112, first paragraph be withdrawn.

35 U.S.C. § 112, Second Paragraph

The Examiner rejected claims 3, 5, 12, 14, 21 and 23 under 35 U.S.C. § 112, second paragraph, as being indefinite for allegedly failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

The Examiner argues: "Claims 3, 5, 12, 14, 21 and 23: The. term "portlet specific method" in claims is a relative term which renders the claim indefinite. The term "portlet specific method" is not defined by the claim, the specification does not provide a standard for ascertaining the requisite degree, and one of ordinary skill in the art would not be reasonably apprised of the scope of the invention."

In response, Applicants respectfully contend that the phrase "portlet specific method" is clearly understood from ordinary usage of the English language to be a method that is different for different portlets. Conversely, a method that is not a "portlet specific method" does not vary from one portlet to another portlet.

Formally, the phrase comprising the suffix "specific" or "-specific" and having the form "N specific" or "N-specific", where N denotes a noun, means specific to the values or species of the noun N. For example something that is "color specific" (i.e., the noun N is "color") varies from one color to another color (i.e., is different for red, blue, green, etc.).

See the website "http://www.mirriamwebster.com/cgi-bin/dictionary" of the Mirriam-Webster's Online Dictionary, which defines "-specific" as "relating or applying specifically to or intended specifically for", and indicates "gender-specific" as an example (meaning specific to the values of gender, namely male and female).

Thus, "portal specific method" is nothing more than a phrase of the English language

which does not have to be defined in a patent application, just as ordinary English language phrases are generally not defined in patent applications.

Applicants could cite hundreds of issued patents having claims that include phrase of the form "N specific" or "N-specific", where N denotes a noun, similar to Applicants' usage such that the phrase having the form "N specific" or "N-specific" is not defined anywhere in the claims or specification. For illustrative purposes, Applicants cite the following three issued patents.

In United States Patent 6,976,247 (issued December 13, 2005 to Altfeld and titled "Generating An Executable File") , claims 8 and 15 recite the limitation "add *a compiler specific index* to the list variable". Neither the claims nor the specification defines "compiler specific index".

In United States Patent 7,079,286 (issued July 18, 2006 to Lapstun et al.and titled "Method of Graphics Imaging") , claim 2 recite the limitation "converted to *device specific CMYK data*". Neither the claims nor the specification defines "device specific CMYK data".

In United States Patent 6,111,893 (issued August 29, 2000 to Voltsun et al.and titled "Universal Protocol Conversion") , claims 3-5, 7-9, and 17-18 recite the limitation *"protocol specific message"* and claims 3-4 recite the limitation *"protocol specific method"*. Neither the claims nor the specification defines "protocol specific message". Neither the claims nor the specification defines "protocol specific method".

The preceding cited patents are merely illustrative and Applicants could cite many more such patents that claim phrases analogous to "portlet specific method".

The preceding cited demonstrate that the use of the phrase "portlet specific method" is consistent with the ordinary practice by the United States Patent and Trademark Office in

allowing claims of numerous patents that include phrases analogous to "portlet specific method".

Accordingly, Applicants respectfully request that the rejections of claims 3, 5, 12, 14, 21 and 23 under 35 U.S.C. § 112, second paragraph be withdrawn.

35 U.S.C. § 101

The Examiner rejected claims 1-9 and 19-27 under 35 U.S.C. § 101 because the claimed invention is allegedly directed to non-statutory subject matter.

Claims 1-9

Claims 1-2 have been canceled and amended claim 3 includes all of the limitations of claims 1-2.

The Examiner argues: "Applicant claims a method for designing object-oriented software for implementing portlets of a portal. However, there are no detail steps in the claims that disclose about how to implement portlets of a portal. It is just an abstract idea rather than a practical application of the idea. So this renders claims 1-9 to be non-statutory."

In response, Applicants have amended independent claim 3 to recite "designing the software followed by storing the designed software in a computer usable medium from which the software may be executed on a processor of a computer system to implement portlets of a portal", resulting in claimed subject matter that is not an abstract idea.

Accordingly, Applicants respectfully request that the rejections of claims 3-9 under 35 U.S.C. § 101 be withdrawn.

Claims 19-27

Claims 19-20 have been canceled and amended claim 21 includes all of the limitations of claims 19-20.

The Examiner argues: "Applicant claims a computer program product, comprising a

computer usable medium having computer readable object-oriented software embodied therein for implementing portlets of a portal. However, because the method of implementing portlets which the applicant claimed is an abstract ideas in claims 1-9 above. So the computer program product which contains the abstract idea is also considered as non-statutory."

In response, Applicants have amended independent claim 19 to recite: "said computer readable object-oriented software containing program code that when executed by a processor of a computer system implements portlets of a portal", resulting in claimed subject matter that is not an abstract idea.

Accordingly, Applicants respectfully request that the rejections of claims 21-27 under 35 U.S.C. § 101 be withdrawn.

35 U.S.C. § 102(b)

The Examiner rejected claims 1-6, 8-15, 17-24 and 26-27 under 35 U.S.C. § 102(b) as allegedly being anticipated by Hanis (Hanis *et al.*, "Applying the State Pattern to WebSphere Portal Portlets", Part 1 - Overview, Part 2 - Implementation, 12/11/2002).

Since claims 1-2, 10-11, and 19-20 have been canceled, the rejection of claims 1-2, 10-11, and 19-20 under 35 U.S.C. § 102(b) is moot.

Applicants respectfully contend that Hanis does not anticipate claims 3, 12, and 21, because Hanis does not teach each and every feature of claims 3, 12, and 21.

For example, Hanis does not teach the feature: "including a portlet code module ... in the program code, wherein the portlet code module is adapted to execute the action performed method ...; and including a BasePortlet class and a TemplatePortlet class in the portlet code module, wherein the BasePortlet class does not include a portlet specific method and includes the action performed method ..., and wherein the TemplatePortlet class is a child of the BasePortlet class and includes at least one portlet specific method" for claim 3 and equivalent language expressing the same features for claims 12 and 21.

The Examiner argues: "Claims 3, 12 and 21: Hanis also discloses the portlet code module (part 1, page2, "StateMangerPortlet") which includes the action Performed method. (Part 1, page 2, "StateMangerPortlet implements the action performed method...")".

In response, Applicants note that the Examiner relies on the teaching of the StateManagerPortlet class in Hanis as allegedly representing the claimed BasePortlet class of claims 3, 12, and 21. Applicants further note that Hanis teaches (part 1, page2, "StateMangerPortlet") that the StateManagerPortlet class "is portlet-independent, and is where

you typically write all of the portlet-specific code." In other words, the StateManagerPortlet class in Hanis includes code that is portlet specific and als 7, 16 and 25 o code that is not portlet specific (i.e., portlet-independent means not portlet specific).

In contrast, the preceding feature of claims 3, 12, and 21 recites that the "BasePortlet class does not include a portlet specific method", which is clearly not satisfied by the StateManagerPortlet class of Hanis since the StateManagerPortlet class "is where you typically write all of the portlet-specific code."

Moreover, claims 3, 12, and 21 recite that the TemplatePortlet class "is a child of the BasePortlet class and includes at least one portlet specific method" which Hanis does not teach, since the StateManagerPortlet class "is where you typically write *all of* the portlet-specific code."

Based on the preceding arguments, Applicants respectfully maintain that Hanis does not anticipate claims 3, 12, and 21 , and that claims 3, 12, and 21 are in condition for allowance. Since claims 4-6, 8, and 9 depend from claim 3, Applicants contend that claims 4-6, 8 and 9 are likewise in condition for allowance. Since claims 13-15, 17, and 18 depend from claim 12, Applicants contend that claims 13-15, 17, and 18 are likewise in condition for allowance. Since claims 22-24, 26, and 27 depend from claim 21, Applicants contend that claims 22-24, 16 and 27 are likewise in condition for allowance.

In addition with respect to claims 4, 13, and 22, Applicants maintain that Hanis does not teach the feature: "an action listener method in the BasePortlet class".

The Examiner argues: "Claims 4, 13 and 22: Hanis further discloses that the action listener class includes an action performed method to respond to user action event. (Part 1, pages 4, part 2,

pages 6-7, implementation examples)".

In response, Applicants cannot find in Hanis (Part 1, pages 4, part 2, pages 6-7, implementation examples) a teaching of "an action listener method in the BasePortlet class". Therefore, Applicants respectfully request that the Examiner indicate the Examiner's reasoning in concluding that Hanis (Part 1, pages 4, part 2, pages 6-7, implementation examples) teaches "an action listener method in the BasePortlet class".

Furthermore with respect to claims 4, 13, and 22, Applicants maintain that Hanis does not teach the feature: "wherein the action object resulting from the user clicking on the link of the first page is communicated from the portal to the action listener method". Applicants note that the Examiner does not even allege that Hanis teaches "wherein the action object resulting from the user clicking on the link of the first page is communicated from the portal to the action listener method".

Accordingly, Hanis does not anticipate claims 4, 13, and 22.

In addition with respect to claims 5, 14, and 23, Applicants maintain that Hanis does not teach the feature: "wherein said designing the software further comprises including a BaseController class and a TemplateControllerForHtml class in the controller code module, wherein the BaseController class does not include a portlet specific method and includes the perform view method, and wherein the TemplateControllerForHtml class is a child of the BaseController class and includes at least one portlet specific method".

The Examiner argues: "Claims 5, 14, and 23: <u>Hanis</u> also discloses that the controller code module("StateMangerProtlet") includes portlet related methods.(Part 1, page 2,

"StatemanagerProtlet implements the action performed method and the doView,doEdit, doHelp, and doConfigure methods.")".

In response, Applicants note that the Examiner relies on the teaching of the StateManagerPortlet class in Hanis as allegedly representing the claimed BaseController class of claims 5, 14, and 23. Applicants further note that Hanis teaches (part 1, page2, "StateMangerPortlet") that the StateManagerPortlet class "is portlet-independent, and is where you typically write all of the portlet-specific code." In other words, the StateManagerPortlet class in Hanis includes code that is portlet specific and also code that is not portlet specific (i.e., portlet-independent means not portlet specific).

In contrast, the preceding feature of claims 5, 14, and 23 recites that "the BaseController class does not include a portlet specific method", which is clearly not satisfied by the StateManagerPortlet class of Hanis since the StateManagerPortlet class "is where you typically write all of the portlet-specific code."

Moreover, claims 5, 14, and 23 recite that the TemplateControllerForHtml class "is a child of the BaseController class and includes at least one portlet specific method" which Hanis does not teach, since the StateManagerPortlet class "is where you typically write *all of* the portlets-specific code.

Furthermore, the Examiner relies on the argument that the StateManagerPortlet class includes both the BasePortlet class and the BaseController class which Hanis does not teach in Hanis (part 1, page2, "StateMangerPortlet").

In further response, Applicants note that the Examiner does not even allege that Hanis

teaches "wherein the BaseController class ... includes the perform view method, as required in claims 5, 14, and 23."

Accordingly, Hanis does not anticipate claims 5, 14, and 23.

In addition with respect to claims 6, 15, and 24, Applicants maintain that Hanis does not teach the feature: "wherein a first object of the State class includes a first perform view method for displaying a first portlets state of a given page, wherein a second object of the State class includes a second perform view method for displaying a second portlets state of the given page, and wherein the first and second portlets states are different portlets states".

The Examiner argues: "Claims 6, 15 and 24: Hanis further discloses that a first object of the state class includes a first perform view method for displaying a first portlets state of a given page and a second object of the State class includes a second perofrmView method for displaying a second portlet state of the given page. (Part 1, page 2, "Typically, the state's perform method will invoke a J.P. to render its results", part 2, page 14-15, section "Wrapping up the portlet states and actions")".

In response, Applicants respectfully contend that the Examiner's citation to Hanis (Part 1, page 2, "Typically, the state's perform method will invoke a J.P. to render its results") does not teach the claimed first perform view method and second perform view method subject to the claimed limitations on the first perform view method and second perform view method.

In further response, Applicants respectfully contend that the Examiner's citation to Hanis (part 2, page 14-15, section "Wrapping up the portlet states and actions") specifically recites:

"The remaining part of the application follows this same pattern. The process control logic flows exactly the same way for the Edit mode as it does for the View mode.

1. The initial state for the Edit view is provided through the InitialStateManager class.

2. The State class's perform view method is invoked, gets its needed model objects, and passes them to the J.P.

3. The J.P. assigns one or more PortletURIs to actions on the UI, and associates each with a specific action class.

4. When the user clicks on a link, the action class's action performed method is invoked. The action logic is performed and the appropriate state is set.

5. Processing continues in this manner as the user navigates through the portlet".

Applicants respectfully contend that the preceding citation to Hanis (part 2, page 14-15, section "Wrapping up the portlet states and actions") does not teach the claimed first perform view method and second perform view method subject to the claimed limitations on the first perform view method and second perform view method.

Accordingly, Hanis does not anticipate claims 6, 15, and 24.

In addition with respect to claims 8, 17, and 26, Applicants maintain that Hanis does not teach the feature: "wherein the software comprises Java software".

The Examiner argues: "Claims 8, 17 and 26: Hanis further discloses that the software comprises Java software. (Part 1, page 3, example of implementation using Java.)".

In response, Applicants respectfully contend that there no disclosure in Hanis that the example in Hanis (Part 1, page 3) uses Java software. The code displayed for this example could

be C++ software. Applicants respectfully request that the Examiner quote specific program code

in this example that allegedly is specific to Java and cannot exist in C++.

Accordingly, Hanis does not anticipate claims 8, 17 and 26.
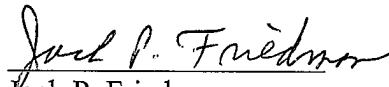
35 U.S.C. § 103(a)

The Examiner rejected claims 7, 16 and 25 under 35 U.S.C. § 103(a) as allegedly being unpatentable over <u>Hanis</u> in view of <u>Hepper</u> (Hepper *et al.*, "Introducing the Portlet Specification, Part 1", 08/01/2003).

Since claims 7, 16 and 25 respectively depend from claims 3, 12, and 21 which Applicants have argued *supra* to not be unpatentable over Hanis under 35 U.S.C. §102(b), Applicants maintain that claims 7, 16 and 25 are likewise not unpatentable over Hanis in view of Hepper under 35 U.S.C. §103(a).

CONCLUSION

Based on the preceding arguments, Applicants respectfully believe that all pending claims and the entire application meet the acceptance criteria for allowance and therefore request favorable action. If the Examiner believes that anything further would be helpful to place the application in better condition for allowance, Applicants invites the Examiner to contact Applicants' representative at the telephone number listed below. The Director is hereby authorized to charge and/or credit Deposit Account 09-0457.

Date: 11/22/2006

Jack P. Friedman
Registration No. 44,688

Schmeiser, Olsen & Watts
22 Century Hill Drive - Suite 302
Latham, New York 12110
(518) 220-1850